22.10.22

BOJ 16956 늑대와 양

문제 요약

- 2차원 격자에 늑대와 양이 있음
- 울타리를 적당히 설치해서 늑대와 양이 같은 공간에 들어가지 않도록 만드는 문제

풀이

- 늑대와 양을 분리할 수 없는 경우를 생각해 보자.
    - 늑대와 양이 인접한 위치에 있으면 울타리로 분리할 수 없음
    - 나머지 경우는 전부 가능할까?
- yes
- 모든 양의 상하좌우를 울타리로 감싸면 됨

BOJ 20309 트리플 소트

문제 요약

- 배열에서 연속한 위치에 있는 세 원소를 선택해서 순서를 뒤집는 연산을 할 수 있다.
- 이 연산만 사용해서 배열을 오름차순으로 정렬할 수 있는지 판별하는 문제

풀이

- 세 원소의 순서를 뒤집는다.
    - $A[i] \leftarrow A[i+2]$
    - $A[i+1] \leftarrow A[i+1]$
    - $A[i+2] \leftarrow A[i]$
    - $A[i]$와 $A[i+2]$의 위치가 바뀜
    - 인덱스의 홀짝이 유지된다는 것을 알 수 있다.
- 홀수 번째 원소와 짝수 번째 원소를 각각 정렬한 다음
- 전체 배열이 정렬되어 있는지 확인하면 됨

BOJ 23656 Jack and Jill

- 이분 탐색 / 상대자 논증 연습 문제

BOJ 9559 Circleland

문제 요약

- i와 $i+1$을 연결하는 길이 L_i짜리 도로, N과 1을 연결하는 길이 L_N짜리 도로가 있음
- 1에서 시작해서 모든 지점을 방문하는데 필요한 최소 이동 거리

풀이

- 이동 경로는 어떤 형태일지 고민해 보자.
- N개의 도로를 모두 사용할 필요가 있을까?
    - 하나를 사용하지 않더라도 모든 지점을 방문할 수 있음
- $N-2$개의 도로만 사용할 수 있을까?

- $N-2$개의 도로를 이용하면 $N-1$개의 지점만 방문할 수 있음
  - 간선이 $N-2$개인 트리의 정점 개수를 생각해 보자.
- 정답으로 가능한 형태
  - 정답은 $N-1$개의 도로를 적당히 방문하는 형태
  - $1-2-3-\cdots-(N-1)-N$
    - $Cost(1, N)$
  - $1-N-(N-1)-\cdots-3-2$
    - $L_N + Cost(2, N)$
  - $1 \to i \to 1 \to N \to (i+1)$
    - $2 \times Cost(1, i) + L_N + Cost(i+1, N)$
  - $1 \to N \to i \to N \to 1 \to (i-1)$
    - $2L_N + 2 \times Cost(i, N) + Cost(1, i-1)$
- $Cost(i, j)$는 $L_i + L_{i+1} + \cdots + L_{j-1}$이므로 누적합 배열 사용

BOJ 5875 오타

문제 요약

- 괄호 문자열이 주어짐
- 문자 하나만 고쳐서 올바른 괄호쌍으로 만드는 경우의 수

풀이

- 올바른 괄호 문자열인지 판별하는 방법
  - 여는 괄호를 +1, 닫는 괄호를 -1로 생각했을 때
  - 합이 0이고 prefix sum이 모두 0 이상이면 됨
- 수식으로 표현
  - i번째 문자가 여는 괄호면 $A[i] = +1$, 닫는 괄호면 $A[i] = -1$로 정의하자.
  - $S[i] = A[1] + A[2] + \cdots + A[i]$라고 정의하자.
  - 어떤 문자열이 올바른 괄호 문자열이라는 것은 $S[N] = 0 \land \min S = 0$이라는 것과 동치이다.
- 여는 괄호를 닫는 괄호로 바꾸는 경우
  - $A[i]$가 2 만큼 감소하므로 $S[i]$부터 $S[N]$까지 모두 2씩 감소
  - $S[N] - 2 = 0$이고 $\min S[1 \cdots i-1] \geq 0, \min S[i \cdots N] - 2 \geq 0$인지 확인하면 됨
- 닫는 괄호를 여는 괄호로 바꾸는 경우
  - $A[i]$가 2 만큼 증가하므로 $S[i]$부터 $S[N]$까지 모두 2씩 증가
  - $S[N] + 2 = 0$이고 $\min S[1 \cdots i-1] \geq 0, \min S[i \cdots N] + 2 \geq 0$인지 확인하면 됨
- S의 prefix min과 suffix min을 관리하면 된다.

코멘트 - 추천 문제

- BOJ 14476 최대공약수 하나 빼기

BOJ 6051 시간 여행

문제 요약

- 3가지 쿼리를 처리하는 문제
  - 스택의 맨 뒤에 x 추가
  - 스택의 맨 뒤에 있는 원소 제거
  - K번째 쿼리를 수행하기 직전으로 돌아감

풀이

- persistent stack을 구현하는 문제
  - persistent data structure: 과거의 상태를 모두 보존하고 있는 자료구조
  - persistent stack: 과거의 상태를 모두 보존하고 있는 스택
  - persistent segment tree: 과거의 상태를 모두 보존하고 있는 세그먼트 트리
- 스택을 연결 리스트로 직접 구현
- 연결리스트의 노드에서는 원소의 값과 바로 밑에 있는 원소의 포인터를 저장
- Top[i] = i번째 쿼리를 처리한 뒤 스택의 맨 뒤에 있는 원소의 포인터를 관리하면 됨

BOJ 25713 괴도 인하

문제 요약

- 격자에 축에 평행한 직사각형 형태의 장애물이 있음
- $[r_{1,i}, r_{2,i}] \times [c_{1,i}, c_{2,i}]$ 장애물은 w_i 만큼의 비용으로 없앨 수 있음
- 오른쪽/아래로만 이동하면서 $(1, 1)$에서 (N, M)으로 가는데 필요한 최소 비용

풀이

- 오른쪽/아래로만 이동할 수 있으므로 같은 장애물의 영역으로 2번 들어가는 경우 없음
  - 각 장애물의 영역에는 한 번 들어가거나 아예 들어가지 않음
  - 들어갔는데 나오지 않을 수도 있음 (도착 지점이 장애물에 포함된 경우)
- 장애물의 영역에 들어갈 때마다 비용을 더하자.
  - 장애물의 위에서 아래로 들어가는 경우
  - 장애물의 왼쪽에서 오른쪽으로 들어가는 경우
  - DP로 계산할 수 있음

BOJ 9520 NP-Hard

문제 요약

- 그래프가 주어지면 N개의 정점을 모두 한 번씩 방문하는 최단 경로를 찾는 문제
- K번 정점을 방문하기 위해서는 $i < K$인 모든 정점 i를 방문한 다음에 K를 방문하거나, 모든 $i < K$를 K 이후에 방문해야 함

풀이

- 조건을 만족하는 방문 순서를 만드는 방법을 생각해 보자.
  - $1, 2, \cdots, N$번 정점을 차례대로 배치하자.
  - 1번 정점을 일단 배치한다.
  - 2번 정점은 맨 앞에 오거나 맨 뒤에 와야 한다.
  - 3번 정점은 맨 앞에 오거나 맨 뒤에 와야 한다.
  - ...
  - 정점의 번호가 감소하다가 1 찍고 증가하는 형태
- $D(i, j) :=$ 맨 앞에 i, 맨 뒤에 j가 있을 때, $\max(i, j) + 1, \cdots, N$번 정점을 추가로 배치하기 위해 필요한 최소 비용
  - $t = \max(i, j) + 1$이라고 하자.
  - t를 맨 앞에 배치하면 $D(i, j) \leftarrow D(x, j) + Cost(x, i)$
  - t를 맨 뒤에 배치하면 $D(i, j) \leftarrow D(i, x) + Cost(j, x)$
  - $O(N^2)$

BOJ 17675 램프

문제 요약

- 0과 1로만 구성된 길이 N짜리 수열 A, B가 주어짐
- 아래 3가지 연산을 이용해서 A를 B로 만들 때 필요한 연산의 최소 횟수
  - 구간 $[l, r]$을 0으로 변경
  - 구간 $[l, r]$을 1로 변경
  - 구간 $[l, r]$의 상태를 반전

풀이

- 기본적인 관찰
  - 한 지점에 1, 2번 연산을 모두 사용할 필요 없음
  - 한 지점에 3번 연산을 여러 번 사용할 필요 없음
  - 3번 연산은 1, 2번 연산을 모두 끝낸 다음에 해도 됨
  - 1, 2번 연산을 적용하는 구간은 서로 겹치지 않음
  - 3번 연산을 적용하는 구간은 서로 겹치지 않음
- 중요한 관찰
  - 1번 연산을 사용한 구간과 2번 연산을 사용한 구간이 인접하지 않는 최적해가 존재
  - 만약 인접하는 최적해가 존재한다면, 두 구간의 합집합에 1번 연산을 적용한 뒤, 2번 연산을 사용할 구간에 3번 연산을 사용하면 됨
- 전략
  - 수열에 0, 1, x를 적당히 깔아놓자. (x는 그대로)
  - xx00xx00xx11xx00xx11 같은 형태
  - 이 상황에서 3번 연산을 적당히 적용하면 됨
- 점화식
  - $D(i, j) := A[1 \cdots i]$만 신경썼을 때, i번째 값에 j를 깔아놓고 $A[1 \cdots i]$를 $B[1 \cdots i]$와 동일하게 만들 때 필요한 연산의 최소 횟수
  - $C(i, j) := i$번째 값에 j를 깔았을 때 $B[i]$와 동일하면 0, 다르면 1 (3번 연산이 필요한지 확인)
  - 상태 $3N$개, 각 상태의 답을 $O(1)$에 구할 수 있으므로 $O(N)$에 해결할 수 있음

코멘트 - 추천 문제

- 1, 2번 연산은 그 전까지의 모든 작업을 덮어버리는 연산이라는 점을 생각하면 풀이를 생각하기 쉬움
- BOJ 11934 Fortune Telling 2

BOJ 18123 평행우주

문제 요약

- $s \le 30$개의 정점으로 구성된 트리 $n \le 10^6$개가 주어짐
- 서로 다른 트리의 개수를 구하는 문제

풀이

- 두 rooted tree T_1, T_2가 위상 동형인지 판별하는 방법
  - 만약 정점이 1개라면 당연히 위상 동형
  - 두 트리의 루트 r_1, r_2가 자식을 갖고 있다면, 각 자식을 루트로 하는 서브트리들이 서로 위상 동형이어야 함
  - 서브 트리를 적당한 기준으로 정렬한 다음, 차례대로 비교하면 됨
  - 적당한 기준은 나중에 알아보자.

- 두 unrooted tree T_1, T_2가 위상 동형인지 판별하는 방법
  - rooted tree의 위상 동형을 판별하는 방법은 간단하니까 이번에도 루트를 고정해 보자.
  - 어떤 정점을 루트로 고정하지?
    - centroid는 항상 1개 또는 2개 존재하고, 2개 존재하면 두 centroid는 인접함
    - 따라서 centroid가 1개면 그 정점을 루트로 고정하고
    - 2개면 두 정점 사이에 정점을 하나 추가한 다음 루트로 고정
- 트리를 정렬하는 기준
  - 트리를 적당한 정수로 변환할 수 있다면 정수를 정렬하는 것은 쉬움
  - 트리의 오일러 투어를 생각해 보자.
  - 자식으로 내려가는 것을 0, 부모로 올라가는 것을 1이라고 하면 트리를 bitstring으로 나타낼 수 있음
  - $|V| \le 31$이므로 64bit integer로 저장할 수 있음
- 또는...
  - 트리를 해싱해도 됨